CLAIMS

What is claimed is:

1. A computer-implemented method for compiling program code, comprising:

generating first object code segments optimized at a first optimization level;

generating second object code segments optimized at a second optimization level, wherein the second object code segments are respectively associated with the first object code segments;

identifying checkpoints in the program code, the checkpoints delineating the object code segments; and

generating checkpoint code for execution at the checkpoints, wherein the checkpoint code saves state information of the program.

2. The method of claim 1, wherein the second optimization level includes no optimizations.

3. The method of claim 2, wherein the first optimization level includes more optimizations than the second optimization level.

4. The method of claim 1, wherein the first optimization level includes more optimizations than the second optimization level.

5. The method of claim 1, further comprising undoing optimizations made in generating the first object code segments in generating the second object code segments.

6. The method of claim 5, further comprising:

identifying checkpoints in the program code, the checkpoints delineating the object code segments; and

generating checkpoint code for execution at the checkpoints, wherein the checkpoint code saves state information of the program.

7. The method of claim 1, further comprising:

generating segments of intermediate code;

for each segment of intermediate code,

optimizing the segment of intermediate code in generating a corresponding one of the first object code segments; and

undoing optimizations of intermediate code in generating a corresponding one of the second object code segments.

8. The method of claim 7, further comprising:

identifying checkpoints in the program code, the checkpoints delineating the object code segments; and

generating checkpoint code for execution at the checkpoints, wherein the checkpoint code saves state information of the program.

9. The method of claim 1, further comprising:

identifying checkpoints in the program code, the checkpoints delineating the object code segments; and

generating checkpoint code for execution at the checkpoints, wherein the checkpoint code saves state information of the program.

10. The method of claim 9, wherein the second optimization level includes no optimizations.

11. The method of claim 10, wherein the first optimization level includes more optimizations than the second optimization level.

12. The method of claim 9, wherein the first optimization level includes more optimizations than the second optimization level.

13. A computer-implemented method for recovery from a program execution error, comprising:

identifying checkpoints in the program code;

generating checkpoint code for execution at the checkpoints, wherein the checkpoint code saves state information of the program;

generating first object code segments optimized at a first optimization level, the object code segments delineated by the checkpoints;

generating second object code segments optimized at a second optimization level, wherein the second object code segments are respectively associated with the first object code segments;

upon detecting a program error, recovering state information of the program from a checkpoint; and

selecting for execution between a first and second object code segment associated with the checkpoint of the recovering step.

14. The method of claim 13, wherein the first optimization level includes more optimizations than the second optimization level, and further comprising:

initially executing the first object code segments; and

retrying execution of a first object code segment associated with the checkpoint from the recovering step before selecting a second object code segment for execution.

15. The method of claim 13, wherein the second optimization level includes no optimizations.

16. The method of claim 15, wherein the first optimization level includes more optimizations than the second optimization level.

17. The method of claim 13, wherein the first optimization level includes more optimizations than the second optimization level.

18. The method of claim 13, further comprising undoing optimizations made in generating the first object code segments in generating the second object code segments.

19. An apparatus for compiling program code, comprising:

means for generating first object code segments optimized at a first optimization level;

means for generating second object code segments optimized at a second optimization level, wherein the second object code segments are respectively associated with the first object code segments;

means for identifying checkpoints in the program code, the checkpoints delineating the object code segments; and

means for generating checkpoint code for execution at the checkpoints, wherein the checkpoint code saves state information of the program.

20. An apparatus for recovery from a program execution error, comprising:

means for identifying checkpoints in the program code;

means for generating checkpoint code for execution at the checkpoints, wherein the checkpoint code saves state information of the program;

means for generating first object code segments optimized at a first optimization level, the object code segments delineated by the checkpoints;

means for generating second object code segments optimized at a second optimization level, wherein the second object code segments are respectively associated with the first object code segments;

means for upon detecting a program error, recovering state information of the program from a checkpoint; and

means for selecting for execution between a first and second object code segment associated with the checkpoint of the recovering step.

21. A computer program product configured for causing a computer to perform the steps of:

generating first object code segments optimized at a first optimization level;

generating second object code segments optimized at a second optimization level, wherein the second object code segments are respectively associated with the first object code segments;

identifying checkpoints in the program code, the checkpoints delineating the object code segments; and

generating checkpoint code for execution at the checkpoints, wherein the checkpoint code saves state information of the program.

22. A computer-implemented method for compiling program code, comprising:

generating during compilation of the program code first machine code segments optimized at a first optimization level; and generating during compilation of the program code second machine code segments optimized at a second optimization level, wherein the second machine code segments are respectively associated with the first machine code segments.

23. The method of claim 22, wherein the second optimization level includes no optimizations.

24. The method of claim 23, wherein the first optimization level includes more optimizations than the second optimization level.

25. The method of claim 22, wherein the first optimization level includes more optimizations than the second optimization level.

26. The method of claim 22, further comprising undoing optimizations made in generating the first machine code segments in generating the second machine code segments.

27. The method of claim 26, further comprising:

identifying checkpoints in the program code, the checkpoints delineating the object code segments; and

generating checkpoint code for execution at the checkpoints, wherein the checkpoint code saves state information of the program.

28. The method of claim 22, further comprising:

generating segments of intermediate code;

for each segment of intermediate code,

optimizing the segment of intermediate code in generating a corresponding one of the first machine code segments; and

undoing optimizations of intermediate code in generating a corresponding one of the second machine code segments.

29. An apparatus for compiling program code, comprising:

means for generating during compilation of the program code first machine code segments optimized at a first optimization level; and means for generating during compilation of the program code second machine code segments optimized at a second optimization level, wherein the second machine code segments are respectively associated with the first machine code segments.

30. An article of manufacture, comprising:

a computer-readable medium configured with instructions for causing a processor-based arrangement to perform the steps of,

generating during compilation of the program code first machine code segments optimized at a first optimization level; and

generating during compilation of the program code second machine code segments optimized at a second optimization level, wherein the second machine code segments are respectively associated with the first machine code segments.